ATLAS
A Practical FPGA-based Framework for Novel CMP Research

Sewook Wee, Jared Casper, **Njuguna Njoroge**, Yuriy Teslyar, Daxia Ge,
Christos Kozyrakis, and Kunle Olukotun

Transactional Coherence and Consistency (TCC) Group
Computer Systems Lab
Stanford University
http://tcc.stanford.edu

Challenges of Parallel Programming

- In the era of Chip Multiprocessors (CMPs)
  - Adoption of multicore chips (Intel, AMD, Sun, IBM, etc.)

- Programming for CMPs
  - Currently → Lock-based multi-threaded model
  - Trade-off between performance and correctness
    - Fine-grain locking, improved performance, more bug prone
    - Coarse-grain locking, easier to code, less performance

Transactional Memory to the rescue...

- Transactional memory → easier to program
  - Coarse grain transactions → similar performance to fine-grain locking
- Stanford's TCC, one of many TM proposals
  - Requires HW support for better performance
    - Stores transactional state in data caches

Limitations of Software Simulators

- New technology like TM evaluated using SW simulator
  - Fast enough for micro-benchmarks, small apps, etc.

- BUT much slower for multiprocessors
  - More details (CPUs, CPU-network, caches, etc)
  - Larger apps, datasets, OS, more I/O

- Questionable reliability of result
  - For speed, simulator abstract details away

FPGAs rescuing slow simulators...

- FPGAs systems can make viable platforms for MP research
  - Fast → Can operate > 100 MHz
  - More logic, memory and I/O's
  - Larger libraries of pre-designed IP cores
- Mapping TCC model on FPGAs → ATLAS
  - Inspired by trends in SW simulator + FPGAs
  - We provide feedback to FPGA-skeptics

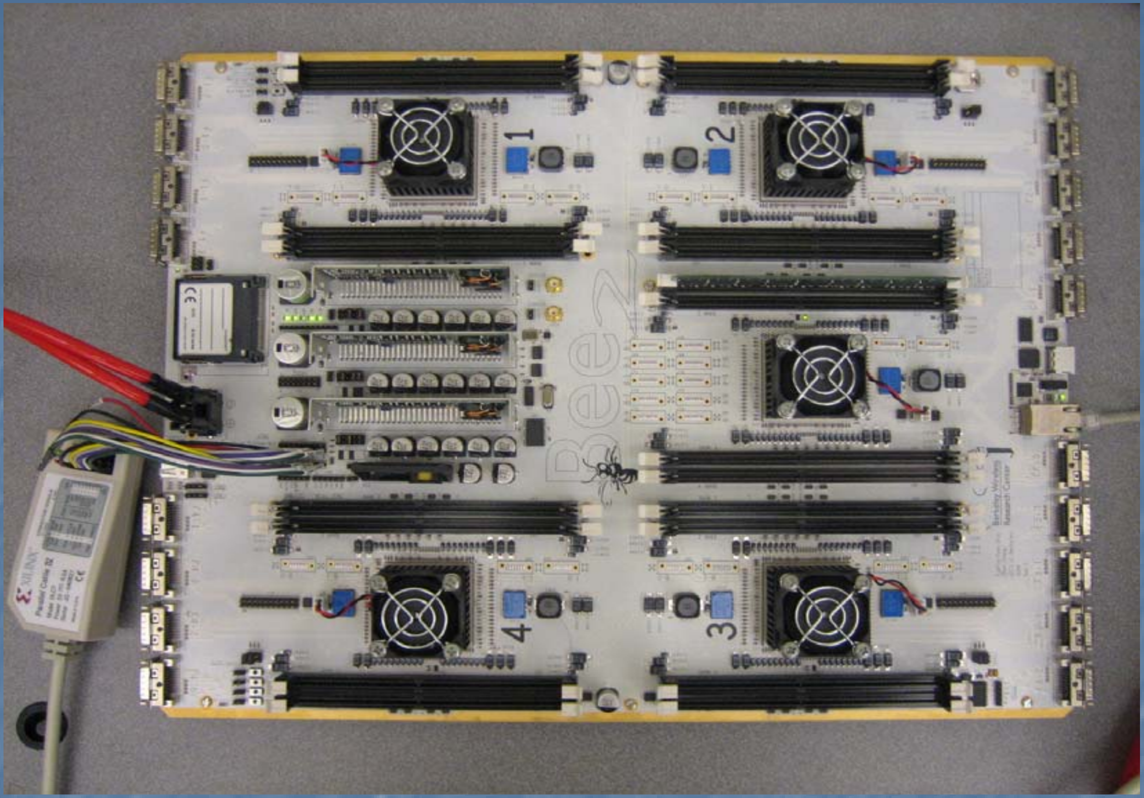
Introducing ATLAS...

- *First* FPGA-based TM CMP
  - Primary objective – accelerate TM software research

- Knocking-out “two birds with one stone”
  - TCC protocol in data caches → Making parallel programming *easier*
  - Mapped onto FPGAs → TM research *faster*

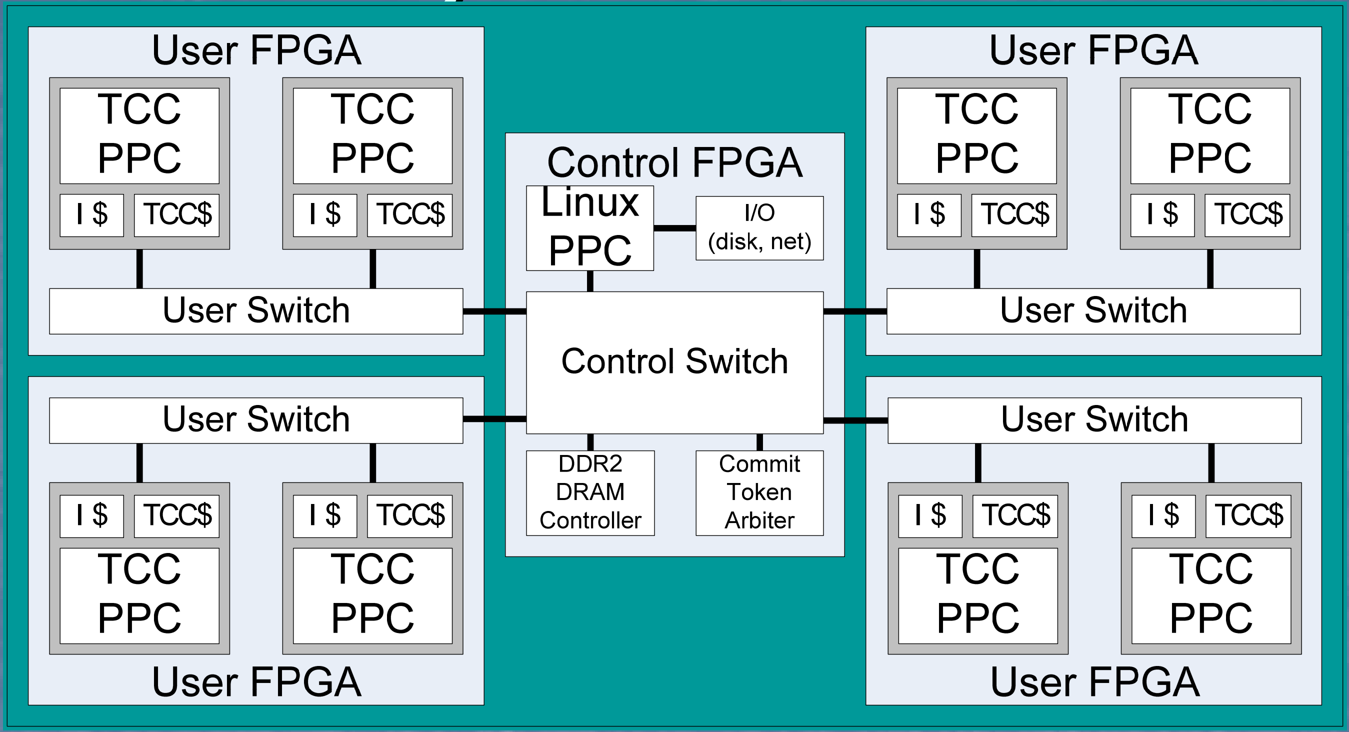
- ATLAS is member of RAMP project
  - RAMP = Research Accelerator for Multiple Processors
  - Consortium of 6 Universities + Industry partners
  - Uses Berkeley Emulation Engine 2 (BEE2) multi-FPGA board
  - ATLAS = *First* working project out of RAMP

The BEE2 Board



- 5 Virtex 2 Pro 70 (~70 K LUTs)
  - 4 *User* FPGAs, direct links to *Control* FPGA
  - 2 embedded PowerPC 405’s per FPGA
  - Each FPGA has access to 2 GB of DDR2 mem
- Rich variety of I/O
  - Ethernet Controllers, Serial interface, etc.

ATLAS 8-way CMP on BEE2 Board



User FPGAs

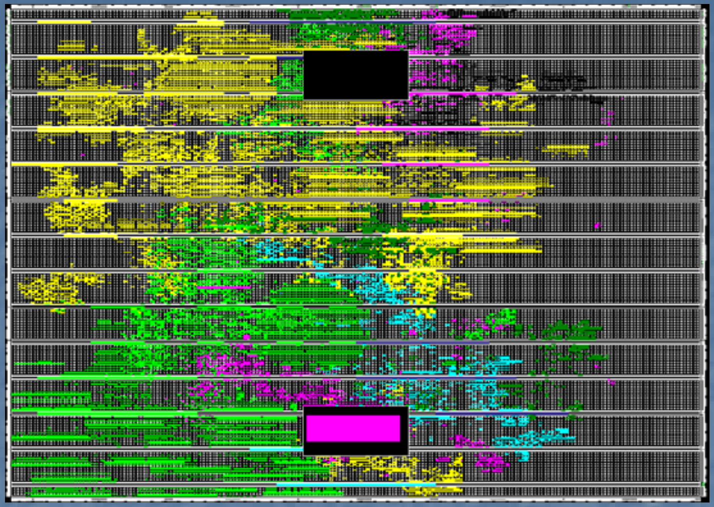
- 4 FPGAs for a total of 8 TCC CPUs
- PPC, TCC caches, BRAMs and busses run @ 100 MHz

Control FPGA

- Linux PPC @ 300 MHz
  - Launch TCC apps here
  - Handle system services for TCC PowerPCs
- Fabric runs @ 100 MHz

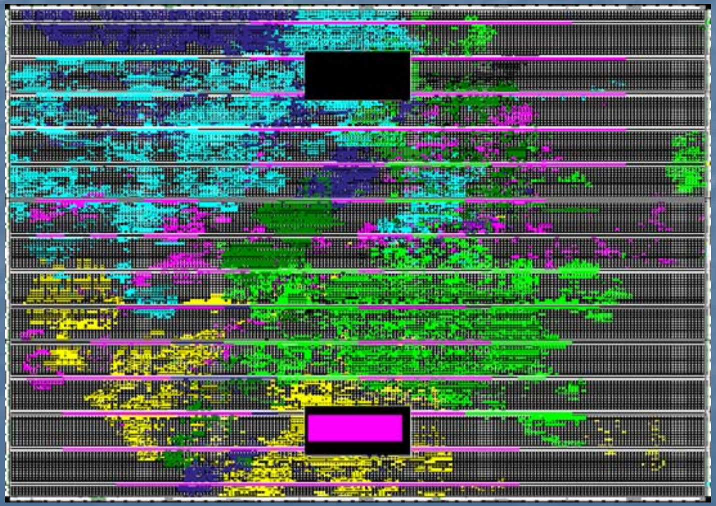
Resource Utilization

- User FPGA (XC2VP70)
  - 17,641 LUTs (26%), 212 KB of BRAMs (32%)
  - Our IP
    - 2x TCC cache = ~14,000 LUTs, 100 KB BRAMs
    - 1x User Switch =~1,700 LUTs, 16 KB BRAMs



Each 45 min. for P & R (ISE 8.1i)

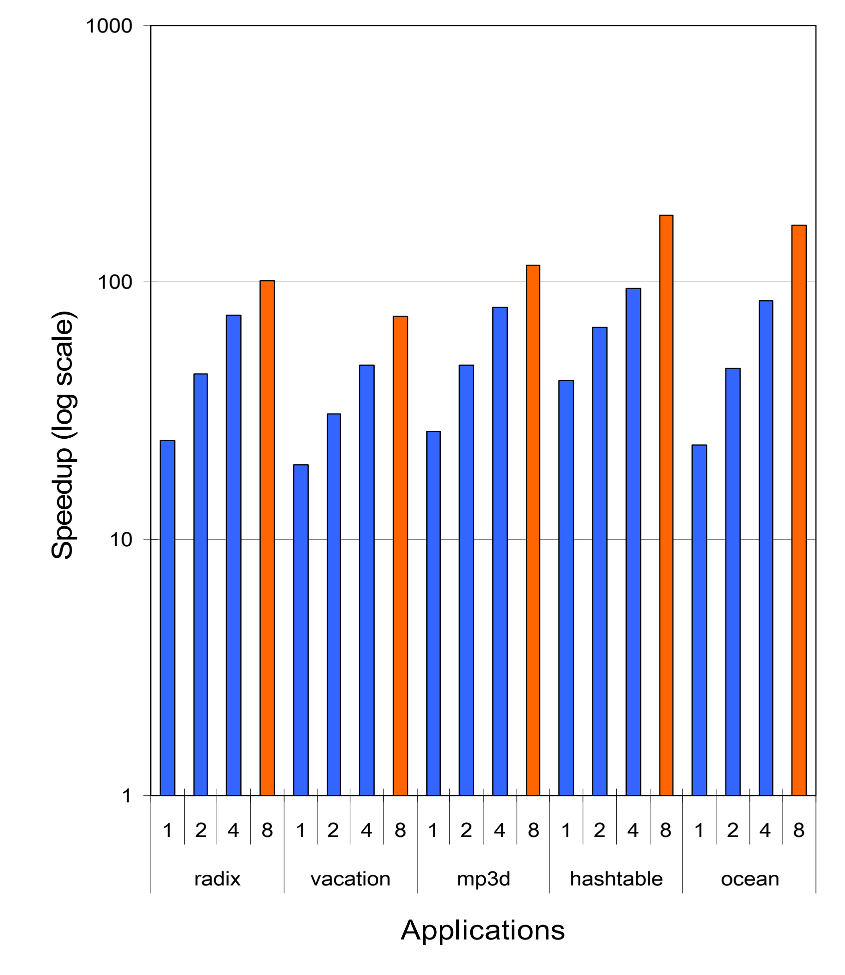
- Control FPGA (XC2VP70)
  - 16,284 LUTs (24%), 66 KB BRAMs (10%)
  - Our IP
    - Control Switch = ~4,800 LUTs, 64 KB BRAMs
    - TCC Arbiter = ~900 LUTs, 4 KB BRAMs



Lessons Learned from ATLAS

- Lesson 1 – Speed advantage of FPGAs
- Lesson 2 – Challenges of mapping ASIC-style RTL on FPGAs
- Lesson 3 – Criteria for choosing base CPU
- Lesson 4 – Using pre-designed IP cores

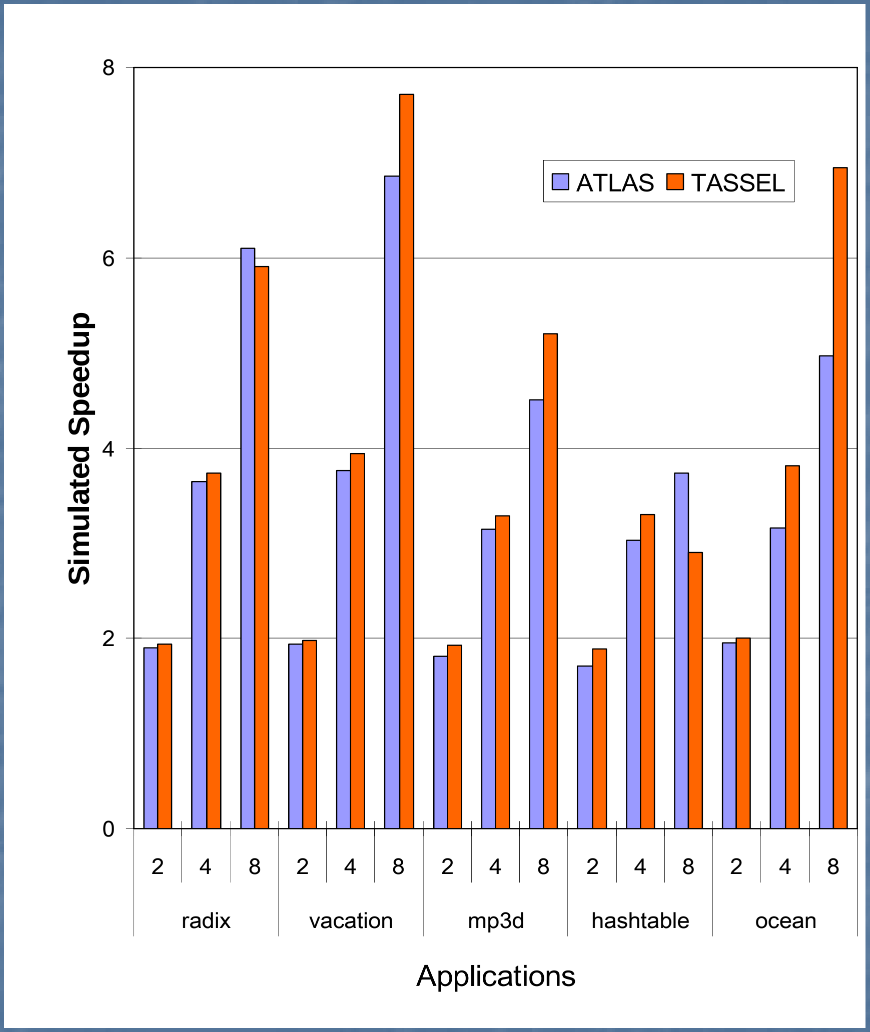
#1) CMPs on FPGAs can outperform SW sim ...



ATLAS Speed advantage over Simulator

- Y-axis = Speedup advantage, log scale
- TASSEL (TCC SW sim)
  - Similar architectural configuration
  - No OS
  - Runs on 2.4 GHz workstation
- 100 MHz ATLAS is on average *124x* faster at 8 processors
- Trends favors FPGA platforms as CMPs increase CPU cores

#1) ... but don't comprise accuracy



ATLAS vs. TASSEL Simulated Speedup

- Y-axis = Self-relative speedup to 1 CPU's <u>simulated</u> execution time

- ATLAS's speedup trends similar to TASSEL's

#2) Mapping certain memory structures and large designs is challenging ...

- Memory structures with gang operations
  - Example: 1-cycle gang clear of cache bits → 160% of LUTs on FPGA
  - Our fix: BRAMs, but multi-cycle clearing → complicated logic
  - Great if BRAMs supported single-cycle clears

- Cannot fit lower-level caches
  - Typical L2 caches in CMPs > 1 MB, but SRAMs on FPGA < 1 MB
  - Our fix: No L2, just use DRAM as main memory

- Designs that span multiple FPGAs
  - Interconnection network (ICN) constrained by board layout
    - Inflexibility of choices in topologies, latency and bandwidth fixed
  - Possible workarounds -- Virtualize ICN

#3) Research with heavy SW-focus...

- ... needs a base CPU with rich SW support
  - Runs full OS like Linux, Solaris, etc. (needs MMU)
  - Large community of users → bountiful programs, debug and profiling tools

- What led us to PowerPC hardcore
  - ATLAS is software research platform
  - Memory hierarchy is focal point of innovation
    - No need to modify processor core, use hardcore, save LUTs for cache

#2 & 3) ATLAS vs. TASSEL differences

Component/Op	TASSEL	ATLAS
Gang-clear	1 cycle	257 cycles
Register Checkpointing of CPU registers	1 cycle	~100 cycles
L1 Hit Time	1 cycle	~10 cycles
Floating Point Ops	< 10 cycles	100's cycles
Interconnection Topology	Shared-bus	Star

Despite differences, scalability trends of apps similar

- Gang-clears, checkpointing occur infrequently
- FP not problem if in parallel regions
- L1 hit time is okay since speedup trends are self-relative

#4) Shared IP cores should have debug interfaces

- Access to libraries of pre-defined IP cores vastly shortens “time-to-market”
  - Only designed 5 out of 30 unique IP cores in ATLAS
  - Allow focus to remain on research-specific cores
- BUT, lack visibility
  - debugging and performance tuning difficult
- Suggestions for improvement
  - Integrate debugging + performance interfaces
  - Examples
    - Performance: Bus contention counters/monitors
    - Debugging: ILA ports to capture & control during debug

Conclusions

- Key Lessons Learned
  - #1) FPGAs are faster, while maintaining accuracy
  - #2) Certain memories and large designs pose challenges
  - #3) SW research requires CPU rich SW support
  - #4) Leverage pre-designed IP cores, improve interfaces
- Multiprocessor research on FPGAs looks promising
  - SW sims are slower
  - Bigger, better FPGAs (BEE2 → BEE3 w/ Virtex 5's)
  - Lesson 4 probably most important for FPGA adoption
- Future of ATLAS
  - Evaluate more TM apps, accelerate our group's TM research

Thanks!

Questions?

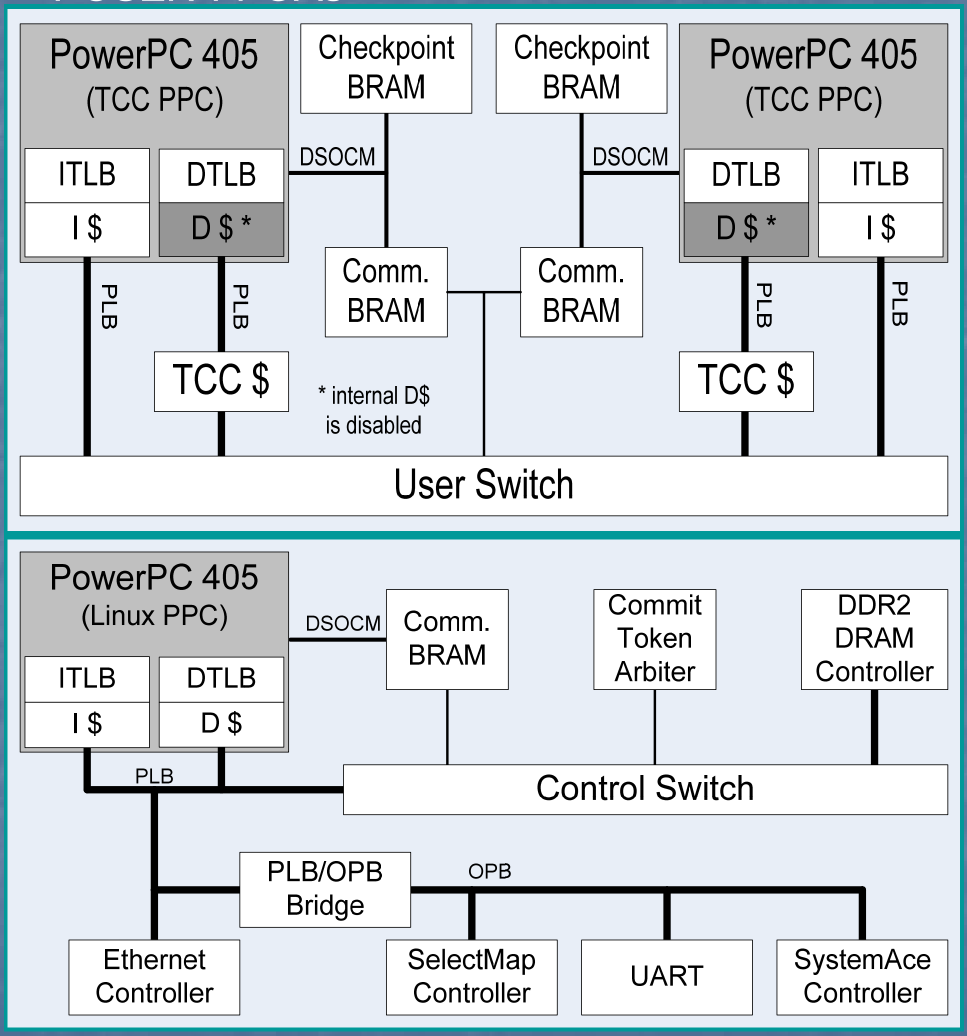
Njuguna Njoroge

tcc_fpga_xtreme@mailman.stanford.edu

http://tcc.stanford.edu/prototypes

ATLAS Hardware Layout

4 USER FPGAs

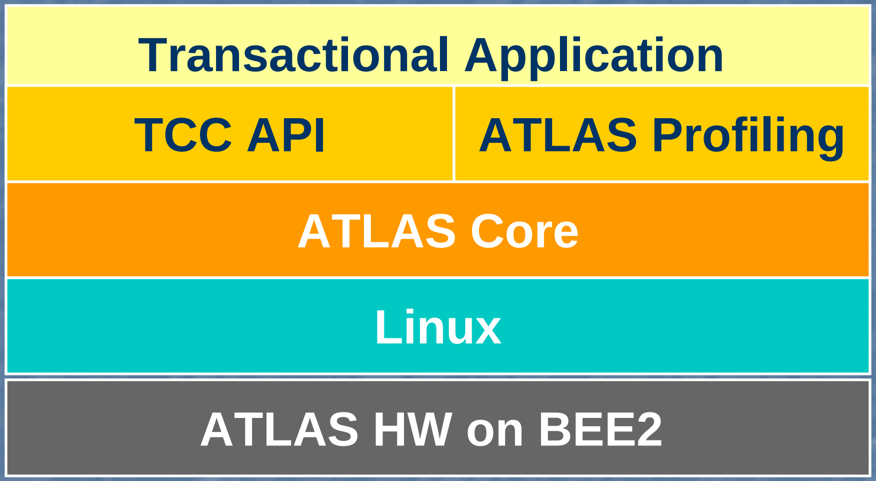


Control FPGA

- User FPGAs
  - 4 FPGAs for a total of 8 TCC CPUs
  - PowerPC, TCC caches, BRAMs and busses run at 100 MHz
- Control FPGA
  - Linux PPC runs at 300 MHz
  - Switch fabric and peripherals run at 100 MHz
  - Links between FPGAs run at 100 MHz
  - 512 MB DDR2 DRAM @ 200 MHz for main mem

ATLAS Software

Software Stack



Transactional Application

TCC API

ATLAS Profiling

ATLAS Core

Linux

ATLAS HW on BEE2

- TCC API – User-visible C functions to define transaction boundaries
- ATLAS Profiling – Gathers stats for performance tuning
- ATLAS Core – Coordinates communication between Linux PowerPC and TCC PowerPCs
- Linux OS (ver 2.4)
  - Full-featured OS
  - All syscall/exception processing on Linux control processor